第 3 章　程序的流程控制

【本章导读】

- 课程内容：程序基本结构、顺序结构、选择结构（if 语句、switch 语句）、循环结构（while 语句、do while 语句、for 语句）、其他控制结构（goto、break、continue、exit()、return）。
- 教学目标：了解结构化程序设计方法以及 3 种基本程序结构；理解顺序结构的程序执行过程；掌握赋值语句；理解选择结构程序设计的概念；理解语句嵌套的概念；掌握循环语句 for 语句、while 语句、do…while 语句、break 语句、continue 语句；学会利用 3 种基本结构进行简单的程序设计。

3.1　程序基本结构

C++程序是由一个或多个函数组成的，每个函数由若干条语句组成，语句是完成具体任务的命令。编写和设计程序其实就是用函数和语句描述和解决实际的问题。描述的过程就是编写程序和组织程序的过程。

目前设计程序有面向过程和面向对象两种主要方法。结构化程序设计是这两种方法的基础和原则。结构化程序设计方法中模块是基本概念，模块化设计是其中心。一个模块可以是一条语句、一段程序、一个函数等。模块相对独立，又可以相互联系。

结构化程序设计把程序模块分为顺序结构、选择结构和循环结构。任何复杂的程序都是由这 3 种基本结构组成的。

顺序结构是程序设计中最简单、最基本的结构，其特点是程序运行时按语句书写的次序依次执行，其结构如图 3-1 所示。

选择结构是根据判定所给的条件是否满足，从而决定程序执行哪些语句，其结构如图 3-2 所示。

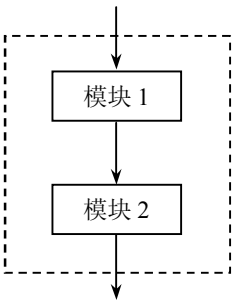

图 3-1　顺序结构流程图

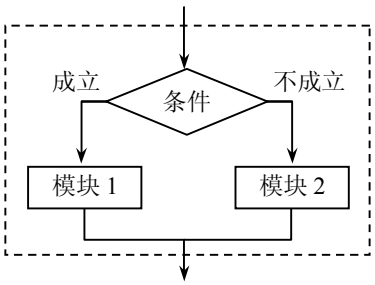

图 3-2　选择结构流程图

循环结构是在满足循环条件的情况下，重复执行循环体模块，当条件不满足时，退出循环模块。循环可分为当型循环和直到型循环，二者的区别在于直到型循环至少执行一次循环体模块，如图 3-3 和图 3-4 所示。

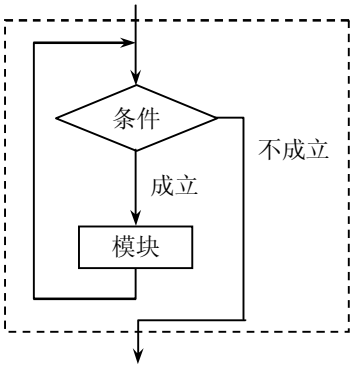

图 3-3　当型循环流程图

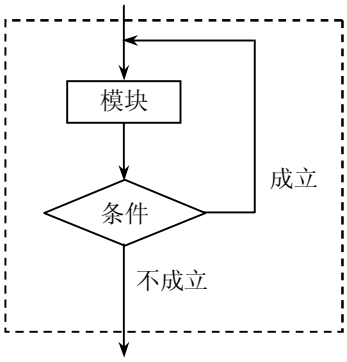

图 3-4　直到型循环流程图

3.2　顺序结构

顺序结构非常简单，将语句顺序排列即可。例如：

```
t=a;
a=b;
b=t;
```

语句执行的顺序就是它们排列的顺序。当语句放大成其他结构时，执行的顺序也是顺序的，例如：

```
a=b;
其他结构
b=c;
…
```

其他结构将在 a=b;后，b=c;前执行。

3.3　选择结构

选择结构又称为分支结构。C++用 if 语句和 switch 语句实现分支结构。

3.3.1　if 语句

if 语句是实现分支结构的主要形式，其基本格式如下：

```
if(<表达式>)
   模块 1;
else
   模块 2;
```

模块可以是一条语句、复合语句，甚至是其他结构。

表达式通常是一个条件判断，用来控制程序的流向。表达式是真（非 0），则执行模块 1，否则执行模块 2。

如果省略 else，则双分支结构变成单分支结构，如：

```
if(<表达式>)
  模块;
```

【例 3.1】输入出租车类型和里程，计算打车的费用。

```
//Example e3_1.cpp
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
    int taxiType;
    float s;
    float money;
    cout << "输入出租车类型（0 或 1）：";
    cin >> taxiType;
    cout << "输入里程：";
    cin >> s;
    if(s < 3 )
        money = 6;   //3 公里以内 6 元
    else
        if( taxiType == 0)
            money = 6 + (s-3)*1.5;     //车型 0 每公里 1.5 元
        else
            money = 6 + (s-3)*1.2;     //其他车型每公里 1.2 元

    cout <<"打车的费用为："<<money<<endl;

    return 0;
}
```

运行结果如图 3-5 所示。

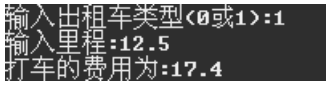

图 3-5　例 3.1 的运行结果

以上程序存在分支结构的嵌套，一个分支结构作为另一个分支结构的分支模块。

读者可以考虑以下情况：

（1）不同车型的起步价不同，如何处理？

（2）打车费用通常是四舍五入，如何处理？

3.3.2　switch 语句

switch 语句又称为多分支开关控制语句，主要用于解决多分支的问题，其格式如下：

```
switch(<整型表达式>)
{
  case <数值 1>: 模块 1;
  case <数值 2>: 模块 2;
  …
  case <数值 n>: 模块 n;
  default: 模块 n+1;
}
```

switch 中的整型表达式和 case 后面的数值形成对应关系，如果匹配则执行对应后面的模块。default 用于不能和所有 case 后面的数值匹配的其他情况，也可以省略。

如果执行某个模块完成后没有跳出 switch 语句，程序将按照顺序执行下一个 case 后面的模块。所以，对于正常的多分支开关控制语句，常常在模块后面加上一句 break;，用于执行后跳出。

多分支开关控制语句的流程图如图 3-6 所示。

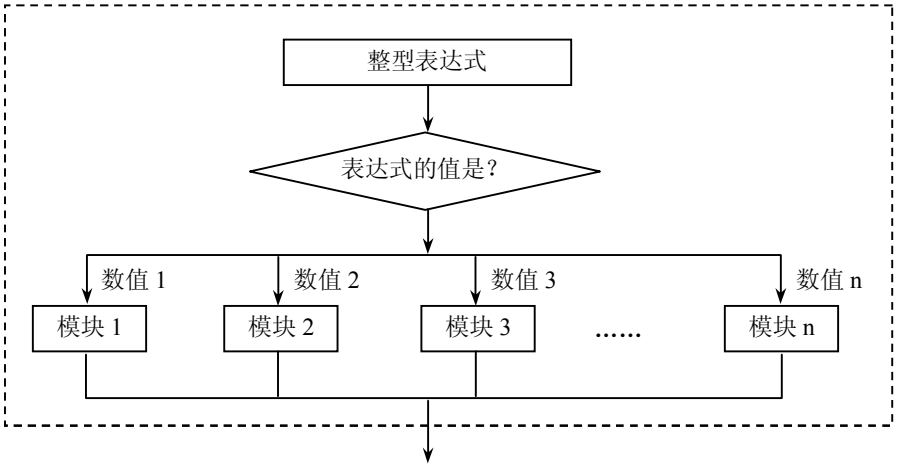

图 3-6　switch 分支语句流程图

【例 3.2】输入一个成绩，判断成绩级别。

```
//Example e3_2.cpp
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
      int score;
```

```
    cout << "请输入成绩：";
    cin >> score;

    switch(score/10)
    {
        case 0:
        case 1:
        case 2:
        case 3:
        case 4:
        case 5:cout<<"不合格";break;
        case 6:
        case 7:cout<<"合格";break;
        case 8:cout<<"良好";break;
        case 9:
        case 10:cout<<"优秀";break;
        default:cout<<"成绩输入可能有问题";break;
    }

    cout<<endl;
    return 0;
}
```

运行结果如图 3-7 所示。

图 3-7　例 3.2 的运行结果

score 为 80，则 switch 后面的整型表达式 score/10 等于 8，对应 case 8: …，所以输出“良好”。

如果输入 95，则 switch 后面的整型表达式 score/10 等于 9，对应 case 9:，后面对应的模块为空，由于没有 break 语句，则转到 case 10 后面的模块，输出“优秀”后遇到 break;退出，如图 3-8 所示。

图 3-8　例 3.2 的运行结果

3.4　循环结构

C++语言实现循环的方式主要有 3 种：while 语句、do…while 语句和 for 语句。

3.4.1　while 语句

while 语句构成“当型循环”，其语法如下：

while(<表达式>)　< 循环体>

当表达式不为 0 时重复执行循环体；当表达式为 0 时退出循环。如果表达式永远无法为 0，将形成“死循环”。

如何设计表达式和循环体，需要分析实际问题，并将实际问题的重复性和判断条件把握好。例如以下事件：有 100 人向投币箱投币，第 1 人投 1 元，第 2 人投 2 元，……，第 100 人投 100 元。很显然，最后投币箱中的总金额是 1+2+3+…+100，相当于 1～100 这 100 个自然数的和。计算机用变量和常量处理数据，因此可以设 s 代表投币箱中的金额，i 表示每次投币金额，凑巧的是，每次投币的金额和投币顺序号一样，每次投币操作都可以认为是下面两个操作的重复：

```
s = s + i;          // 向投币箱投入 i 元
i = i +1;           // 第 i +1 人准备投 i +1 元
```

这种投币操作是有限次数的，共 100 次，即 i 超过 100 就停止，写成循环就是：

```
i = 1;              // 第 1 个人准备投 1 元
s = 0;
while(i <=100)
{
  s = s+ i;
  i = i +1;
}
```

【例 3.3】计算 1+2+3+…+100。

```
//Example e3_3.cpp
#include <iostream>
using namespace std;
int main()
{
    int i = 1;              //第 1 个人准备投 1 元
    int s = 0;
    while(i <=100)
    {
        s = s+ i;
        i = i +1;
    }
    cout <<"1+2+3+...+100="<<s<<endl;

    return 0;
}
```

运行结果如图 3-9 所示。

```
1+2+3+...+100=5050
```

图 3-9　例 3.3 的运行结果

其实，上面程序中第 i 人正好投币 i 元，如果第 i 人投 2*i 元，则程序就是计算：

$$2 + 4 + 6 + \cdots + 200$$

读者可以考虑计算以下的和：

（1）1+3+5+…+99。

（2）1+1/2+1/3+1/4+…+1/100。

3.4.2 do…while 语句

do while 语句称为直到型循环。所谓直到型循环是指直到不满足什么条件时退出循环。其语法如下：

```
do
{
  <循环体>
}while(<表达式>);
```

上节求和的程序也可以写成：

```
i = 1;
s = 0;
do
{
  s = s + i ;
  i = i +1;
}while(i<=100);
```

效果和前面的一样。

但观察下面的程序，会发现直到型循环有自己的特色。

观察图 3-10 所示的两个程序。左边当型循环结束后 s 等于 0，而右边直到型循环结束后 s 等于 1。这是因为循环条件都不满足，但直到型循环至少执行一次才会遇到循环条件的判断。

```
m=1;                    m=1;
s=0;                    s=0;
while(m>1)              do
{                       {
  s=s+m;                  s=s+m;
  m=m+1;                  m=m+1;
}                       }while(m>1);
```

图 3-10 当型循环和直到型循环的比较

正常情况下，两种循环没什么区别。

3.4.3 for 语句

for 语句是最常用的循环，其语法如下：

```
for(<表达式 1>; <表达式 2>; <表达式 3>) <循环体>
```

循环执行方式为：

表达式 1：只执行一次。通常是为循环作准备。

表达式 2：通常作为循环条件部分，如果为空，相当于真值。

表达式 3：通常作为循环变量调整的部分。

循环体：重复运行的模块。这个部分也可以对循环进行调整和控制。

for 循环执行流程图如图 3-11 所示。

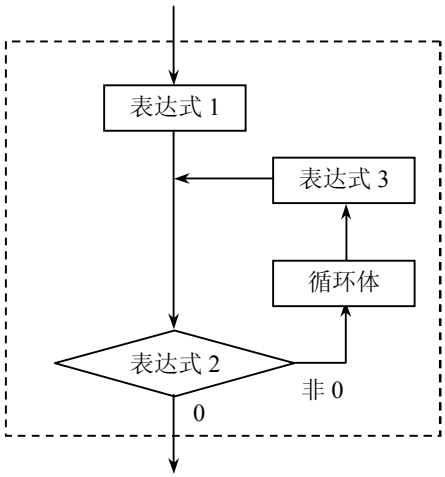

图 3-11　for 循环执行流程图

以上求和的程序用 for 语句可以写成：

```
for(i = 1,s = 0 ; i <=100 ; i++)
    s = s + i;
```

for 语句可以有以下省略方式：

```
for(;②;③)  循环体;
for(;②;)  循环体;       //相当于 while(②)  循环体;
for(;;)  循环体;         //相当于 for(;1;)  循环体;
for(;;);                 //死循环
```

for 语句构造的循环通常称为计数型循环。循环变量控制了整个循环，可以通过循环变量的初值和终值以及每次调整的数值（步长）来计算循环的次数，例如上面的程序的循环次数等于：$\frac{\text{终值-初值}}{\text{步长}}+1$，即 $\frac{100-1}{1}+1=100$。

【例 3.4】输入 10 个数，输出其中最大和最小的数。

```
//Example e3_4.cpp
#include <iostream>
using namespace std;
int main()
{
    int max,min;
    int a;
    cout <<"输入第 1 个数：";
```

```
    cin >> a;
    max = min = a;
    for(int i = 2 ; i <= 10 ; i++)
    {
        cout <<"输入第"<<i<<"个数：";
        cin >> a;
        if( a > max)
            max = a;
        else
            if( a < min)
                min = a;
    }
    cout << "max ="<<max<<endl;
    cout << "min ="<<min<<endl;

    return 0;
}
```

程序运行后按照提示输入 10 个数，结果如图 3-12 所示。

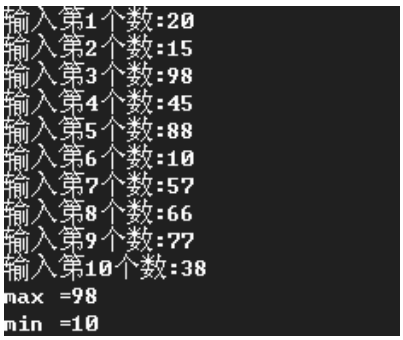

图 3-12　例 3.4 的运行结果

3.4.4　循环的嵌套

和分支结构一样，循环也允许嵌套。嵌套的循环可以完成更复杂的任务。

【例 3.5】计算并输出 2～100 之间的素数之和。

分析：所谓素数指的是除了 1 和自身以外，没有其他因子。根据这个定义，可以用除了 1 和自身以外的小于该数的数去除该数，如果所有的数都不能整除，则可以判断该数是素数。

例如，对于自然数 7，用 2、3、4、5、6 去除 7，发现都不能整除，则判断出 7 是素数；而对于自然数 15，用 2～14 之间的数（包括 2 和 14）去除 15，则发现 3、5 可以整除 15，则 15 不是素数。

程序如下：

```
//Example e3_5.cpp
#include <iostream>
```

```
using namespace std;

int main()
{
    int s=0;
    int i,j;
    for(i = 2 ; i <= 100 ; i++)              //产生 2～100 之间的数 i
    {
        for( j = 2 ; j <= i-1 ; j++)         //产生除了 1 和 i 以外的数
            if( i % j == 0 ) break;          //发现 i 的因子
            if( j ==i )                      //2～i-1 之间没有发现因子，则 i 是素数
                s = s + i;                   //累加素数
    }

    cout << "2～100 之间的素数之和为："<<s<<endl;

    return 0;
}
```

程序运行结果如图 3-13 所示。

```
2～100之间的素数之和为:1060
```

图 3-13　例 3.5 的运行结果

程序中判断素数的循环嵌入在大循环之内，完成每个 i 是否是素数的判断。当所有的 j 都判断完成，没有一个 j 是 i 的因子，则 i 必然是素数。这时候 j 必然不符合循环条件 j<=i-1，由于 j 是递增的，所以这时候 j 其实等于 i。

其实判断素数只要用 2～i/2 之间的数，甚至是 2～sqrt(i)之间的数也可以完成判断的任务。使用 sqrt 需要包含 cmath 头文件。

3.5　其他控制语句

1. break 语句

break 语句用于 switch 语句和循环语句中，不能用于其他语句。其功能是跳出开关控制语句和跳出循环。如例 3.5 中的 break 语句可以跳出 j 循环。

2. continue 语句

与 break 语句退出循环不同的是，continue 语句只结束本次循环，接着进行下一次循环的判断，如果满足循环条件，继续循环，否则退出循环。

【例 3.6】演示跳转语句。

```
//Example e3_6.cpp
#include <iostream>
using namespace std;
```

```
int main()
{
   int i,s;
   for(i=1,s=0;i<=10;i++)
   {
      if(i % 2 == 0)
         continue;
      if(i % 10 == 7)
         break;
      s = s + i;
   }
   cout << "s="<<s<<endl;

   return 0;
}
```

程序运行结果如图 3-14 所示。

```
s=9
Press any key to continue
```

图 3-14　例 3.6 的运行结果

程序分析：程序中当 i 是偶数，即 i%2==0 时，遇到 continue，程序转向 for 语句的表达式 3：i++;，当 i%10==7，即对 10 求余等于 7 时遇到 break 语句，程序将跳出循环。其他的数都会遇到 s = s+i;，所以可以看出，程序遇到 7 将退出循环输出 s。i 从 1 循环到 7，只有 1、3、5 这 3 个数累加到变量 s 中，所以最后 s 的值是 1+3+5 等于 9。

3. goto 语句

goto 语句为无条件转向语句，形式为：

```
goto　语句标号
```

语句标号用标识符表示，命名规则同变量名。例如下面的程序段：

```
i=1;
s=0;
MySub:if （i<=100）
{
   s=s+i;
   i=i+1;
   goto MySub;   // MySub 就是标识符
}
…
```

对于结构化程序设计，不主张使用 goto 语句，否则会导致程序流程混乱、可读性差，一般用在特殊的场合，且不宜多用。

4. exit()函数

exit()是一个库函数，执行该函数将返回操作系统，常用于终止程序的执行。exit()可以返回值，例如：

```
exit(1);
```

5. return

return 也可以完成跳转，通常用于完成任务或异常情况下的返回，具体在函数章节中阐述。

3.6　程序示例

【例 3.7】输出下面的图形：

```
    *
   ***
  *****
 *******
*********
```

程序如下：

```
//Example e3_7.cpp
#include <iostream>
using namespace std;

void main()
{
    int i,j;
    for(i = 1 ; i <= 5 ; i++)
    {
        for( j = 1 ; j <= 5 - i ; j++)
            cout << " ";
        for( j = 1 ; j <= 2*i-1 ; j++)
            cout << "*";
        cout <<endl;
    }
}
```

程序分析：这是一类图形输出的例子。设计这样的程序需要抓住内外循环的关联。

其中需要找到以下规律：

（1）星号前的空格个数与行数之间的关系。

（2）星号个数与行数之间的关系。

对于本题，行号 i 从 1 开始循环到 5，规律如下：

（1）星号前的空格个数为 5-i 个。

（2）星号个数为 2*i-1 个。

【例 3.8】计算 s=1×2×3×4×5×6×7×8。

分析：题目其实就是求 8!。计算阶乘和求和很相似，不过还是有一些区别。

程序如下：

```
//Example e3_8.cpp
#include <iostream>
```

```
using namespace std;

int main()
{
    int t=1;
    for(int i=1 ;i<=8;i++)
        t = t * i;

    cout<<"8!="<<t<<endl;
    return 0;
}
```

运行结果如图 3-15 所示。

```
8!=40320
Press any key to continue
```

图 3-15　例 3.8 的运行结果

【例 3.9】输出 2009 年 1 月的日历。2009 年 1 月 1 日是星期四，要求按照正常的日历排版格式，具体如图 3-16 所示。

```
     2009 年  1 月

日  一  二  三  四  五  六
                1   2   3
 4   5   6   7   8   9  10
11  12  13  14  15  16  17
18  19  20  21  22  23  24
25  26  27  28  29  30  31
```

图 3-16　例 3.9 的运行结果

输出日历格式需要考虑以下因素：

（1）输出月标题、星期标题。

（2）输出 1 月 1 日前面的空格。

（3）换行输出的处理。

假设每天输出占位 4 个字符（前后各一个空格，天数占两个字符），1 月 1 日前面的空格个数应该是 4*4。

换行处理需要考虑 1 月 1 日星期几和是否是 7 的倍数，其判断式为：

```
(week + day)  %  7 == 0
```

具体程序如下：

```
//Example e3_9.cpp
#include <iostream>
#include <cstring>
#include <iomanip>

using namespace std;
```

```
int main()
{
    int year = 2009;
    int month = 1;
    int week = 4;
    int space;
    int day ;
    //每月的标志
    cout << setw(10) << year
         << setw(4) << "年"
         << setw(4) << month
         << setw(4) << "月" << endl <<endl;
    cout << setw(4) << "日"
         << setw(4) << "一"
         << setw(4) << "二"
         << setw(4) << "三"
         << setw(4) << "四"
         << setw(4) << "五"
         << setw(4) << "六" <<endl;
    //输出空格
    for( space = 0 ; space < week ; space ++)
        cout<<"      ";          //每天对应 4 个空格

    for(day = 1 ; day <= 31; day ++)
    {
        cout << setw(4) << day ;
        if ((day +week) % 7 ==0 )
            cout <<endl;
    }
    cout <<endl;

    return 0;
}
```

读者可以考虑如何输出一年的日历表，同时可以考虑如何将两个月或 3 个月的日历横排输出。

【例 3.10】利用循环计算圆的面积。

分析：假设圆的半径为 R，圆的方程是 $x^2+y^2 = R^2$，则 $y = \sqrt{R^2 - x^2}$ 。也可以写成自变量 x 的函数形式：

$$f(x) = \sqrt{R^2 - x^2}$$

写成表达式为：

sqrt(pow(R,2)-pow(x,2))

将 R 分成 n 等份，每一个等份可以看成是一个梯形，梯形的面积可以计算成：

(f(a)+f(b))*h/2

b-a 等于 r/n，即每个梯形的高。将 n 个梯形面积累加起来就可以计算出四分之一大小的圆的面积。实际计算的时候可以将*h/2 最后统一处理，如图 3-17 所示。

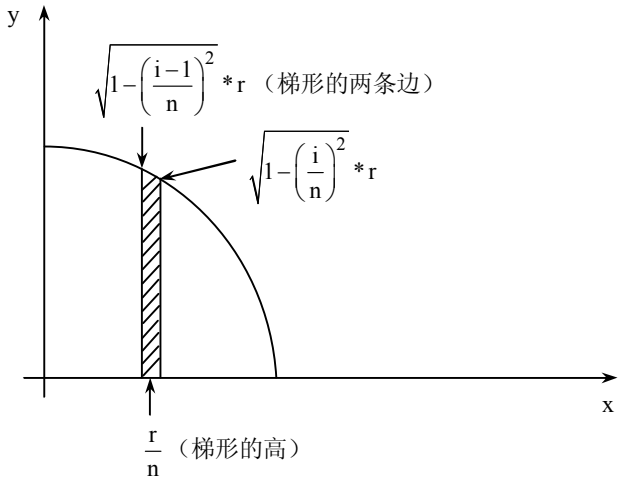

图 3-17　计算圆面积示意图

程序具体如下：

```
//Example e3_10.cpp
#include <iostream>
#include <cstring>
#include <cmath>
#include <iomanip>

using namespace std;

int main()
{
    double area = 0,r=10,h,s_area;
    long n=100000;
    cout << setprecision(8) << "用公式计算的面积为：" << 3.1415926 *10 *10   <<endl;
    h = (double ) r /n;
    for(int i = 0 ; i< n ;i++)
    {
        s_area = ( sqrt(pow(r,2)-pow(i*h,2)) + sqrt(pow(r,2)-pow((i+1)*h,2))) * h / 2 ;
        area = area + s_area;
    }
    cout << setprecision(8) << "用循环计算的面积为：" << area * 4 << endl;

    return 0;
}
```

运行结果如图 3-18 所示。

图 3-18　例 3.10 的运行结果

【例 3.11】利用循环求不定方程的解。

数学中的不定方程是指其中自变量的值不能直接解出，可能有多个解。例如，假设 x、y 都是非负整数，则方程 x+9y=20 的解有：

x=2，y=2

x=11，y=1

x=20，y=0

这 3 种情况。其判断求解的依据不仅是方程的等式，还有自变量是非负整数这个特征限制。

下列问题就是一个典型的不定方程问题。

假设有 20 元，可以由 10 元、5 元和 1 元组成。问可以有哪些组成方式？

设 20 元中 10 元、5 元和 1 元各为 x、y、z 张，则有：

x*10+5*y+z*1 == 20

下面的程序列出了所有的解：

```
//Example e3_10.cpp
#include <iostream>
#include <cstring>

using namespace std;

int main()
{
    int x,y,z;
    cout <<"10 元" <<'\t'
         <<"5 元" <<'\t'
         <<"1 元" <<'\t'<<endl;
    for(x = 0 ; x <= 2; x++)
        for(y = 0 ; y <= 4 ; y++)
            for( z = 0;z <= 20 ;z++)
                if(x*10+y*5+z == 20)
                    cout <<x <<'\t'
                         <<y <<'\t'
                         <<z <<'\t'<<endl;
    return 0;
}
```

运行结果如图 3-19 所示。

【例 3.12】住房贷款公式。

住房贷款的计算方法有两种：等额本息法和等额本金法。

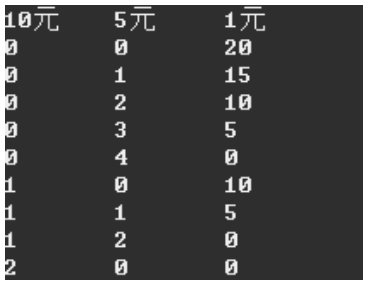

图 3-19 例 3.11 的运行结果

等额本息法：每个月本息总额一样，开始还款主要还利息，其计算公式为：

$$p * rm * (1+rm)^{month} / ((1+rm)^{month} -1)$$

其中，p 为本金，rm 为月利率，month 为总还款月数。

假如贷款 10 万元，分 10 年还清，年利率 7.2%，则 rm = 7.2% /12，month = 120。

等额本金法：每个月还的本金相同，利息随着本金的减少而减少，所以每个月的还款额在不断减少。第 i 个月的还款公式为：

$$p/month +(p - (i-1) * p/month) * rm$$

下面的程序进行分别计算，考虑到篇幅的原因，其中等额本金法只输出第一年的还款情况。

```
//Example e3_12.cpp
#include <iostream>
#include <iomanip>
#include <cstring>
#include <cmath>
using namespace std;

int main()
{
    double p=100000.0;   //本金，即贷款额
    double q,s;
    int year = 10,month = year*12;      //贷款年份和总月数
    double r = 0.0720,rm = r/12;        //年利率和月利率
    q = p * rm * pow(1+rm,month) / (pow(1+rm,month)-1);     //等额本息公式
    cout << setw(6) << "等额本息法：" <<endl;
    cout << setiosflags( ios::fixed | ios::showpoint)
         << setw(8) << "每月还款："
         << setw(9) << setprecision(2)   << q <<endl;
    cout << setw(6) << "总利息："
         << setw(9) << setprecision(2) << q*month - p << endl <<endl;

    cout << setw(6) << "等额本金法：" <<endl;
    s = 0;     //计算累计的利息
    for(int i=1 ; i<=month ; i++)
    {
```

```
            q = p / month +( p - (i-1) * p / month ) * rm;     //等额本金公式
            s = s + q - p / month;
            if(i<=12)
            cout << setiosflags( ios::fixed | ios::showpoint)
                 << setw( 2 ) <<"第"   << setw(2) << i << setw(8) << "月还款："
                 << setw( 9 ) << setprecision(2) << q <<endl;

        }
        cout << setw(6) << "总利息："
             << setw(9) << setprecision(2) << s <<endl;
        return 0;
    }
```

运行结果如图 3-20 所示。

图 3-20　例 3.12 的运行结果

【例 3.13】验证 9999 是否符合“歌德巴赫猜想”。

歌德巴赫（Goldbach C.，1690.3.18～1764.11.20）是德国数学家，出生于格奥尼格斯别尔格（现名加里宁城），曾在英国牛津大学学习，原学法学，由于在欧洲各国访问期间结识了贝努利家族，所以对数学研究产生了兴趣，曾担任中学教师。

“歌德巴赫猜想”是歌德巴赫在 1742 年 6 月 7 日给著名数学家欧拉的信中提出的一个命题：

随便取某一个奇数，比如 77，可以把它写成 3 个素数之和：

$$77=53+17+7$$

再比如 461：

$$461=449+7+5$$

下面用循环来验证 9999 是否符合这个猜想。

```
//Example e3_13.cpp
#include <iostream>
```

```
#include <cstring>
using namespace std;

bool isprimer(int);

int main()
{
    int n=9999;
    int a,b,c;
    for(a = 2 ;a < n ;a++)
    {
        if(isprimer(a))
            for( b = 2;b < n ;b++)
            {
                c = n - a - b ;
                if(isprimer(b) &&  isprimer(c))
                {
                    cout << "符合歌德巴赫猜想"<<endl;
                    cout << n << "="
                        << a << "+"
                        << b << "+"
                        << c << endl;
                    exit(0);
                }
            }
    }
    cout << "不符合歌德巴赫猜想"<<endl;
    return 1;
}
bool isprimer(int n)   // 判断 n 是否是素数
{
    for(int i = 2 ; i <= n/2 ; i++)
        if(n % i == 0) break;
        if(i > n/2 ) return true;
        else return false;
}
```

程序运行结果如图 3-21 所示。

图 3-21　例 3.13 的运行结果

【例 3.14】输出九九乘法表。

九九乘法表要求输出数字时控制好宽度。

```
//Example e3_14.cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int i,j;
    for(i=1;i<=9;i++)
    {
        for(j=1;j<=i;j++)
            cout << setiosflags( ios::fixed)
                 << setw(1) << i
                 << setw(1) << "*"
                 << setw(1) << j
                 << setw(1) << "="
                 << setw(2) << i*j <<" ";
        cout<<endl;
    }
    return 0;
}
```

运行结果如图 3-22 所示。

```
1*1= 1
2*1= 2 2*2= 4
3*1= 3 3*2= 6 3*3= 9
4*1= 4 4*2= 8 4*3=12 4*4=16
5*1= 5 5*2=10 5*3=15 5*4=20 5*5=25
6*1= 6 6*2=12 6*3=18 6*4=24 6*5=30 6*6=36
7*1= 7 7*2=14 7*3=21 7*4=28 7*5=35 7*6=42 7*7=49
8*1= 8 8*2=16 8*3=24 8*4=32 8*5=40 8*6=48 8*7=56 8*8=64
9*1= 9 9*2=18 9*3=27 9*4=36 9*5=45 9*6=54 9*7=63 9*8=72 9*9=81
```

图 3-22　例 3.14 的运行结果

程序中输出 i*j 的时候设置 2 位宽度。每一个 j 循环结束后都需要换行。

【例 3.15】计算 s=1-2+3-4+5-…-100

程序其实可以直接口算出来，但为了一般性，下面的程序设计了通用的解决方案。

```
//Example e3_15.cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int a,b,c;
    int flag = 1;
```

```
    a=b=c=0;      //赋值运算符的结合方向是从左向右
    for(int i = 1 ; i <= 100 ; i++)
    {
        //1．利用 flag 改变符号
        a = a + flag * i;
        flag = - flag ;
        //2．利用 i 的属性
        b = (i%2 == 1) ? b+i:b-i;
        //3．利用 i 的属性
        c = c + (i%2 == 1) ? i : -i;
    }
    cout <<"1-2+3-...+99-100 = " << a << endl;
    cout <<"1-2+3-...+99-100 = " << b << endl;
    cout <<"1-2+3-...+99-100 = " << c << endl;

    return 0;
}
```

运行结果如图 3-23 所示。

```
1-2+3-...+99-100 = -50
1-2+3-...+99-100 = -50
1-2+3-...+99-100 = 100
Press any key to continue
```

图 3-23　例 3.15 的运行结果

注意到 c 的输出和预期的不一致，这是因为以下语句：

```
c = c + (i%2 == 1) ? i : -i;
```

其实相当于：

```
c =(c + (i%2 == 1)) ? i : -i;
```

所以条件运算符“?:”的优先级要低于“+”。

本章小结

（1）结构化程序设计方法。3 种基本程序结构：顺序结构、选择结构和循环结构。

（2）选择结构是根据某种条件的成立与否而执行不同的程序模块。选择结构又可分为简单分支（两个分支）和多分支两种情况。一般采用 if 语句实现简单分支结构程序，用 switch 和 break 语句实现多分支结构程序。if 语句主要有 4 种句式：单分支的 if 语句、双分支的 if 语句、多分支的 if 语句和嵌套的 if 语句。

（3）循环结构是 3 种结构中最重要的一种结构。循环结构包括 while、do…while 和 for 循环（goto 也可以构成循环，通常不用）。while 循环和 do…while 循环的条件判断一个在前，一个在后，为导致循环体执行的次数不同，需要密切注意。循环结构的实质是重复执行一个程序模块，但这种重复性是在循环条件的有效控制之下完成的，目的是完成指定的任务，所以利用循环结构设计程序的关键就在于如何控制循环的条件，在恰当的时机由“真”变“假”，从

而退出循环。

（4）break 语句和 continue 语句可以改变循环运行的方向，主要用于特殊情况的处理，但不能控制 if 和 goto 构成的循环。

习题三

一、选择题

1．若有 int a,b,t;，下面不能实现将 a 和 b 中的数据进行交换的是（　）。

A．a=a+b,b=a-b,a=a-b;　　B．t=a,a=b,b=t;

C．a=t; t=b; b=a;　　D．t=b; b=a; a=t;

2．以下合法的 C++语言赋值语句是（　）。

A．a=b=10　　B．s = a+b

C．a=4,b=5　　D．++i;

3．与 y=(x>0?1:x<0?-1:0);的功能相同的 if 语句是（　）。

A．
```
if (x>0) y=1;
  else if(x<0)y=-1;
    else y=0;
```

B．
```
if(x)
  if(x>0)y=1;
  else if(x<0)y=-1;
    else y=0;
```

C．
```
y=-1;
if(x)
if(x>0)y=1;
  else if(x==0)y=0;
    else y=-1;
```

D．
```
y=0;
 if(x>=0)
 if(x>0)y=1;
   else y=-1;
```

4．若有定义 float f;　int a, b;，则合法的 switch 语句是（　）。

A．
```
switch(f)
{
   case 1.0: cout<<"*\n";
   case 2.0: cout<<"**\n";
}
```

B．
```
switch(a);
{
   case 1 cout<<"*\n";
   case 2 cout<<"**\n";
}
```

C．
```
switch(b)
{
   case 1:   cout<<"*\n";
   default:   cout<<"\n";
   case 1+2: cout<<"**\n";
}
```

D．
```
switch(b)
{
   case 1: cout<<"*\n"
   case 2: cout<<"**\n"
   default: cout<<"\n"
}
```

5．有如下程序：

```
#include <iostream>
#include <cstring>
```

```
using namespace std;
int main()
{
   int m=1,a=0,b=0;
   switch(m)
   {
       case 0: b++;
       case 1: a++;
       case 2: a++;b++;
   }
   cout <<"a=" <<a<<",b="<<b<<endl;
   return 0;
}
```

该程序的输出结果是（　）。

A．a=2,b=1　　B．a=1,b=1

C．a=1,b=0　　D．a=2,b=2

6．若变量 c 为 char 类型，能正确判断出 c 为大写字母的表达式是（　）。

A．'A'<=c<='Z'　　B．(c>='A') || (c<='Z')

C．('A'<=c) and ('Z'>=c)　　D．(c>='A')&&(c<='Z')

7．设 a、b、c、d、m、n 均为整型变量，且 a=5，b=7，c=3，d=8，m=2，n=2，则逻辑表达式(m=a>b)&&(n=c>d)运算后，n 的值为（　）。

A．0　　B．1　　C．2　　D．3

8．以下程序的输出结果是（　）。

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int a=1,b;
   b=a-1;
   if(b>a-2)
      cout <<a <<endl;
   else
      if(b<a+2)
         cout <<b<<endl;
      else
         cout <<-a<<endl;
   return 0;
}
```

A．0　　B．-1　　C．1　　D．都不对

9．运行下面的程序：

```
#include <iostream>
#include <cstring>
```

```
using namespace std;
int main()
{
   int a;
   cin >>a;
   if(a>30) cout <<a <<endl;
   if(a>20) cout <<a <<endl;
   if(a>10) cout <<a <<endl;
   return 0;
}
```

若从键盘输入 38↙，则输出结果是（　）。

A．383838　　B．3838　　C．38　　D．3

10．运行下面的程序：

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int a=16,b=21,m=0;
   switch(a%3)
   {
      case 0:m++;break;
      case 1:m++;
      switch(b%2)
      {
         default:m++;
         case 0:m++;break;
      }
   }
   cout <<m<<endl;
   return 0;
}
```

则输出结果是（　）。

A．1　　B．2　　C．3　　D．4

11．for(i=0;i<100;i++);结束后，i 的值是（　）。

A．99　　B．100　　C．101　　D．102

12．下面的程序段循环次数是（　）。

```
int   i=0;
while(i<10)
{
   if(i < 1)
      continue;
```

```
    if(i==5)
      break;
    i++;
  }
```

A．5　　B．6

C．4　　D．死循环，不能确定次数

13．要使下列程序段计算 1+2+3+…+10 的和，则在下划线处填入正确的数是（　）。

```
for(i=0;i<=____ ;i++)
  s = s + i+1;
```

A．9　　B．10　　C．18　　D．20

14．运行下面的程序：

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
  int i=10,j=0;
  do
  {
    j=j+i;
    i--;
  }while(i>5);
  cout <<j <<endl;
  return 0;
}
```

则输出结果是（　）。

A．45　　B．40　　C．34　　D．55

15．运行下面的程序：

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
  int   k=0,a=1;
  while(k<10)
  {
    for(;;)
    {
      if((k%10)==0)
        break;
      else
        k--;
```

```
        }
        k+=11;
        a+=k;
    }
    cout <<k<<a<<endl;
    return 0;
}
```

则输出结果是（　）。

A．21 32　　B．21 33　　C．11 12　　D．10 11

16．以下叙述正确的是（　）。

A．do…while 语句构成的循环不能用其他语句构成的循环代替

B．do…while 语句构成的循环只能用 break 语句退出

C．用 do…while 语句构成的循环，在 while 后的表达式为非零时结束循环

D．用 do…while 语句构成的循环，在 while 后的表达式为零时结束循环

17．有如下程序：

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    int x=3;
    do
    {
        cout <<x--;
    }while(!x);
    return 0;
}
```

该程序的执行结果是（　）。

A．3　2　1　　B．2　1　0　　C．3　　D．2

18．若 i 为整型变量，则下列 while 循环执行的次数为（　）。

```
i=10;
while(i==0) i = i-1;
```

A．0 次　　B．1 次　　C．10 次　　D．无限次

19．下面有关 for 循环的正确描述是（　）。

A．for 循环只能用于循环次数已经确定的情况

B．for 循环是先执行循环体语句，后判断表达式

C．在 for 循环中，不能用 break 语句跳出循环体

D．for 循环的循环体语句中，可以包含多条语句，但必须用花括号括起来

20．以下循环体的执行次数是（　）。

```
int i,j;
for(i=0 , j=3;i <= j ; i+=2 , j--)
    …
```

A．3　　B．2　　C．1　　D．0

21．以下程序的输出结果是（　）。

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int a,b;
   for(a=1,b=1;a<=100;a++)
   {
      if(b>=10)
         break;
      if(b%3==1)
      {
         b+=3;
         continue;
      }
   }
   cout <<a<<endl;
}
```

A．101　　B．3　　C．4　　D．5

二、填空题

1．复合语句在语法上被认为是_____。空语句的形式是_____。C++语言中，语句结束的标志是_____。

2．if (!x) a=3;语句中的!x 可以改写为_____，使其功能不变。

3．判断 a 既是 3 的倍数又是 7 的倍数的表达式是_____。

4．下列程序的运行结果为_____。

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int i, j, s=0;
   for ( i=1, j=5; i<j; i++, j-- )
      s+=i*10+j;
   cout <<s<<endl;
   return 0;
}
```

5．下列程序段的运行结果为_____。

```
#include <iostream>
#include <cstring>
```

```
using namespace std;
int main()
{
   int i=10, s=0;
   for ( ; --i; )
      if ( i%3==0)
        s+=i;
   s++;
   cout <<s<<endl;
   return 0;
}
```

6．下面的程序运行后，a 的值为_____。

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int i, j,a=0;
   for ( i=0; i<2; i++ ) a++;
      for ( j=4; j>=0; j-- ) a++;
}
```

7．下列程序的运行结果为_____。

```
int i=1, s=3;
do
{
   s+=i++;
   if (s%7==0)
      continue;
   else
      ++i;
} while (s<15);
cout <<i<<endl;
```

8．当运行以下程序时，从键盘输入 Hello#<回车>，则下列程序的运行结果是_____。

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
   int v1=0, v2=0;
   char c;
   while ( (c=getchar( ))!= '#')   // getchar()获得用户从键盘输入的一个字符
   {
      switch (c)
```

```
        {
          case 'e':
          case 'o':
          default : v1++;
          case 'l': v2++;
        }
      }
      cout << v1<< ','<<v2;
      return 0;
    }
```

三、编程题

1．输入 3 个整数 a、b、c，编程交换它们的值，即把 a 中的值给 b，把 b 中的值给 c，把 c 中的值给 a。

2．设计一个简单的计算器程序，用户输入运算数和运算符（+、-、*、/、%），输出计算的结果。

3．编写程序，从键盘输入学生成绩，输出对应的等级（100 分为 A，90～99 为 B，80～89 为 C，70～79 为 D，60～69 为 E，小于 60 为 F）。

4．求 1-1/2+1/3-1/4+1/5-1/6+1/7+…+1/99-1/100 的值。

5．计算 1～100 以内所有含 6 的数的和。

6．输出所有的三位水仙花数。所谓水仙花数是指所有位的数字的立方之和等于该数，例如：

$$153=1^3+5^3+3^3$$

7．编写程序输出下面的图形：

```
1
23
456
7890
```

8．编写程序输出下面的图形：

```
    *********
     *******
      *****
       ***
        *
```